Supporting SCA Applications in a Lightweight CCM Environment

Frank Pilhofer
fp@mc.com
Contents

- SCA Evolution
  - Current state of the SCA
  - Leveraging commercial technologies
  - A scenario for a future SCA

- Migrating Waveforms
  - Metadata
  - Resources

- Summary
SCA Evolution

The Ultimate Performance Machine
SCA History

- SCA pioneered component-based development in embedded systems
  - Branched from CCM during finalization
  - Added important concepts of its own

- OMG specifications are catching up, exceeding SCA functionality
  - Lightweight CCM, Streams for CCM, Lightweight Log, Lightweight Services, D+C

- Combine OMG and JTRS efforts in component-based embedded system development
Leverage OMG standardization efforts
Leverage existing specifications
Increase COTS Content in SCA
  - Commercial, not DOD or SDR specific
Focus on Software Radio domain-specific aspects
Future SCA Assumptions:

- SCA Resources become CCM Components
  - Commercially available Component Model
  - Make use of future extensions, e.g., Streams for CCM
- Use of D+C metadata and infrastructure for the deployment of applications
  - More powerful assembly and deployment model
- No changes to Core Framework interfaces

Future SCA Impact:

- Container/Component API changes
- Metadata (SCA Domain Profile) changes
Premise
- SCA Evolution by embracing commercial standards is beneficial for both JTRS and OMG

Adressing Evolution Issues
- Mercury project to study and resolve evolution and migration issues
- Idea: study migration now, so that it will be feasible and not troublesome later
- Resulted in whitepapers and this presentation
SCA Evolution Issues

- Investments into SCA-based infrastructure must be protected
  - Core Framework implementations
  - Applications (Waveforms)
  - Clients (HCIs)
  - Devices

- Application and HCI investments most critical
  - Limited set of “off the shelf” Core Framework implementations and Devices
Migrating Waveforms
Goal:
- Run existing SCA waveforms, unmodified, in a (Lightweight) CCM- and D+C-based environment

Approach:
- Automatic transformation of application metadata, so that application can be deployed by COTS (not SCA or SDR specific) D+C based infrastructure
- Automatic generation of implementation wrappers, so that resources can be executed as components in a CCM Container
Application Metadata Transformation
Strong correlation between SCA Domain Profile and D+C meta-data

Transformation is well-defined (by design)
● SCA Software Assembly Descriptor is transformed to a D+C Component Package, containing a single assembly-based implementation
Mercury whitepaper compared SCA vs. D+C metadata:

- D+C metadata is superset of SCA
- In the process, discovered and resolved a few issues
  - E.g., “devicethatloadedthiscomponentref” resolved via a port delegation mechanism

- All SCA application metadata can be converted to D+C application metadata
Application Implementation Wrappers
SCA Resource Wrapper

- Wrap SCA Resources as a CCM Component
  - So that they can be deployed in a CCM Container
  - Wrapper acts as CCM component, delegating all behavior to Resource implementation

- No performance impact
  - Involved in connection setup, not in data transport

- Can be generated automatically
  - Using port and property names from Software Component Descriptor (CCD)
Alternative: “Executable Device” compatible Node Managers

- SCA Executable Device implementing D+C Node Manager interfaces
- Capable of running Resources “natively” (in addition to CCM components)
- Disadvantage: requires modification of many Node Managers, becoming SCA specific
Summary
Adopting OMG specifications within the SCA has benefits:

- Greater standards base and implementation choice
- More powerful assembly and deployment model
- Combined efforts for future evolution of component-based development
- Make SCA software radio specific -- no need to define a generic infrastructure

Migration issues can be overcome:

- SCA Applications can be migrated to D+C using a one-time, automated process